UC Davis Cluster 1

Maya A. Natalie L. Riley B. Srishti G.

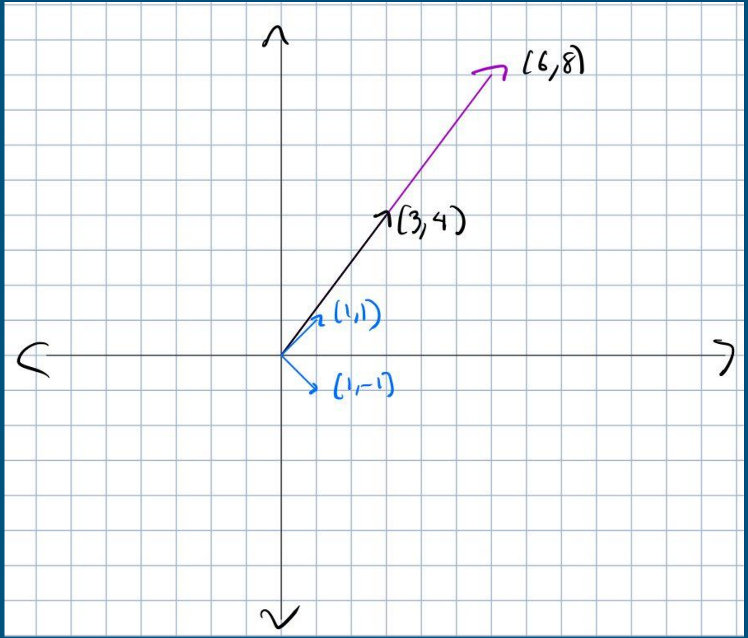
Energy Band Project

Introduction

- Matrix of energy levels
- Each electron in any atom has what we call an energy level - it’s like an address where they live for each electron!
- In this case, we made a matrix to lay out all the possible ‘addresses’ of a series of electrons in a solid

- Matrix: array of numbers
- Multiply matrix x vector = change in magnitude and/or direction of vector
- Eigenvectors: change magnitude but not direction when multiplied by a certain matrix
- Eigenvalue: factor by which magnitude changes when eigenvector is multiplied by the matrix

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} ax + by + cz \\ dx + ey + fz \\ gx + hy + iz \end{bmatrix}$$



$$A = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}, we\ want:$$

$$A\boldsymbol{v} = \lambda \boldsymbol{v}$$

$$det(A - \lambda I) = 0$$

$$det \begin{bmatrix} 1-\lambda & 2 \\ 2 & 1-\lambda \end{bmatrix} = 0 \rightarrow (1-\lambda)^2 - 4 = 0 \rightarrow (\lambda - 3)(\lambda + 1) = 0$$

Formula to find eigenvalues and eigenvectors, in which λ represents eigenvalues

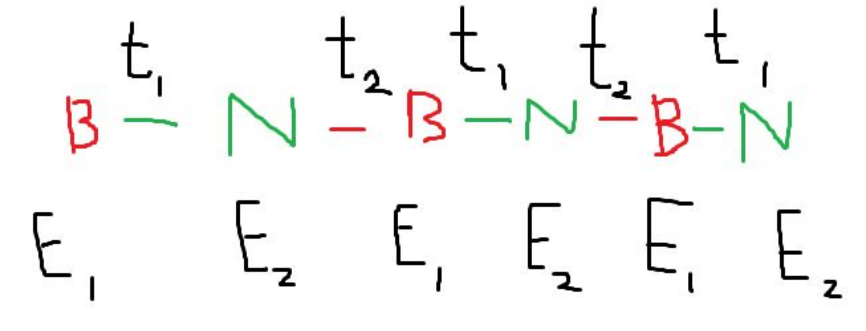
Hamiltonian Matrices & Eigenvalues

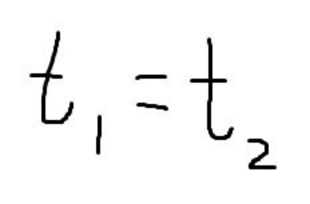
Hamiltonian Matrix + How to construct it:

Hamiltonian Matrix: A matrix whose eigenvalues give the energy levels of the atoms in the solid together

1. Create and label a diagram
2. Find the t; the hopping parameter - the ease in which the e- will have to move between atoms
3. Add the energy levels on the diagonal and ts surrounding it

ex.

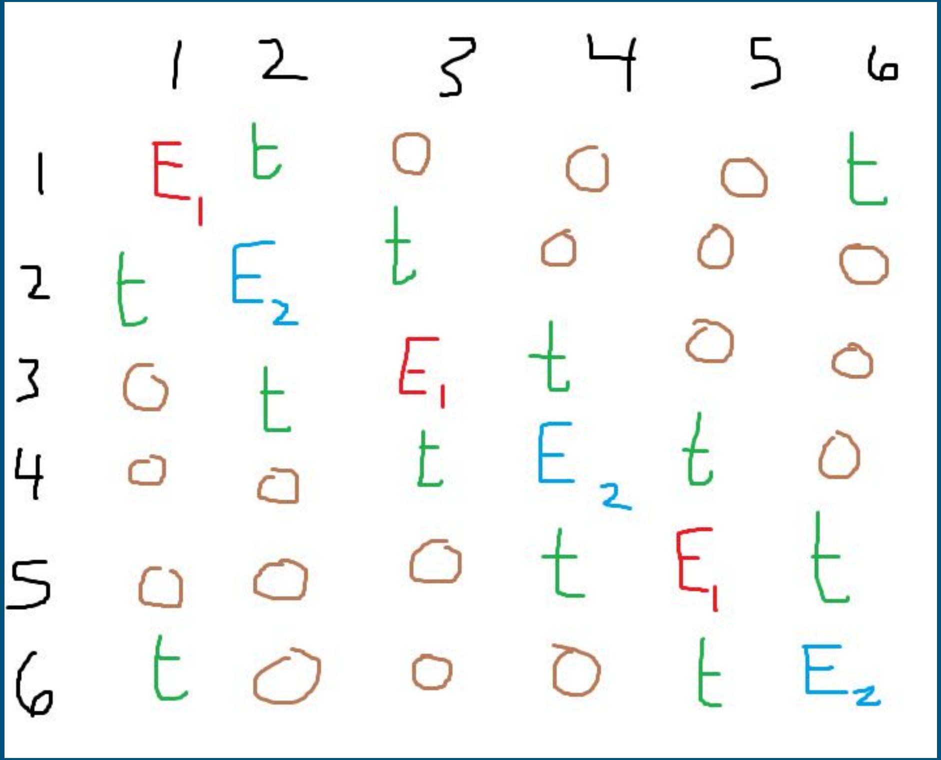




$t_1 = t_2$

Because the bonds are the same length (single-bond N-B)

The constructed Matrix:



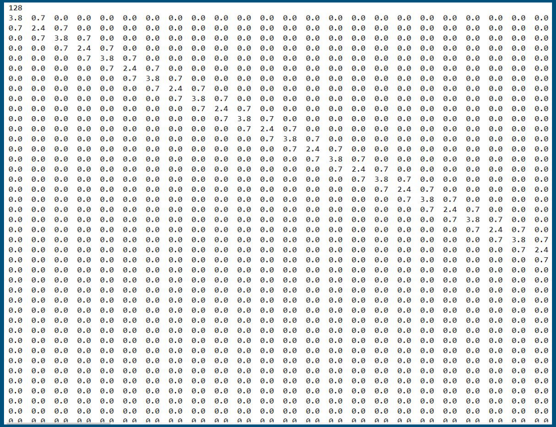
$$\begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline 1 & E_1 & t & 0 & 0 & 0 & t \\ 2 & t & E_2 & t & 0 & 0 & 0 \\ 3 & 0 & t & E_1 & t & 0 & 0 \\ 4 & 0 & 0 & t & E_2 & t & 0 \\ 5 & 0 & 0 & 0 & t & E_1 & t \\ 6 & t & 0 & 0 & 0 & t & E_2 \end{array}$$

Now, we use C to input E1, E2, t1, and t2 (t1=t2=t) and fill out the matrix

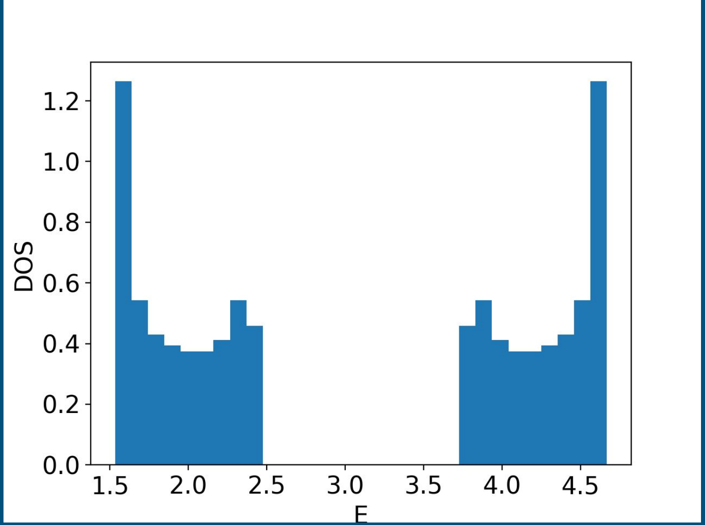
Eigenvalues: the energies of the individual atoms now that they’re banded together

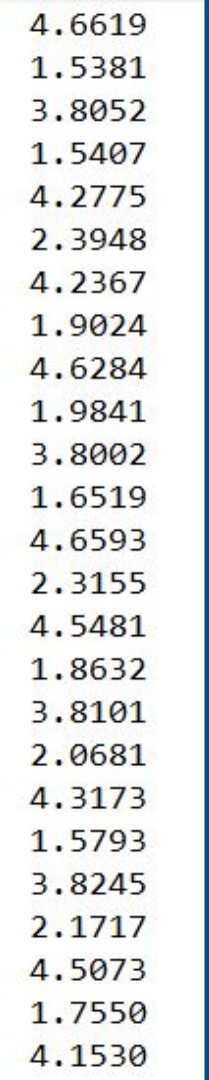
3 Steps in the Code

1. Create the matrix in C
2. Find the eigenvalues in C
3. Use Python to create a histogram of the eigenvalues

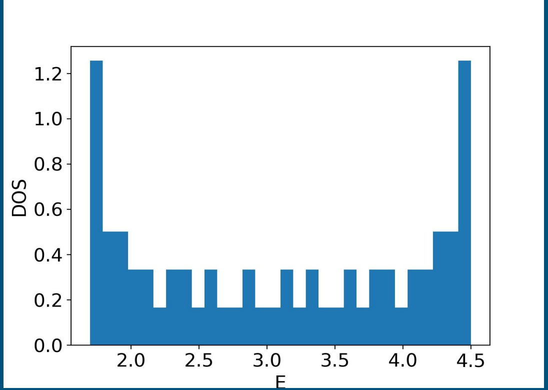


```
128
3.8  0.7  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.7  2.4  0.7  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.7  3.8  0.7  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.7  2.4  0.7  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.7  3.8  0.7  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.7  2.4  0.7  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.7  3.8  0.7  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.7  2.4  0.7  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.7  3.8  0.7  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.7  2.4  0.7  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.7  3.8  0.7  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.7  2.4  0.7  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.7  3.8  0.7  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.7  2.4  0.7  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.7  3.8  0.7  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.7  2.4  0.7  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.7  3.8  0.7  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.7  2.4  0.7  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.7  3.8  0.7  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.7  2.4  0.7  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.7  3.8  0.7  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.7  2.4  0.7  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.7  3.8  0.7
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.7  2.4
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.7
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
```

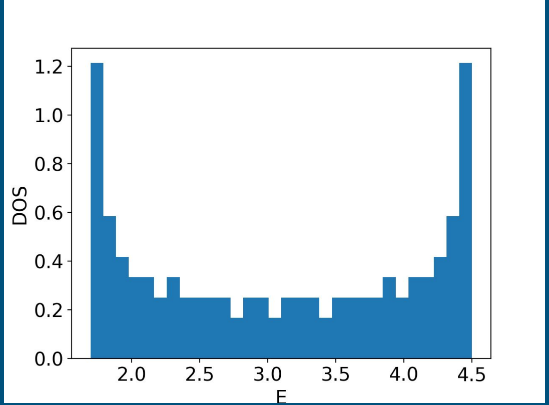




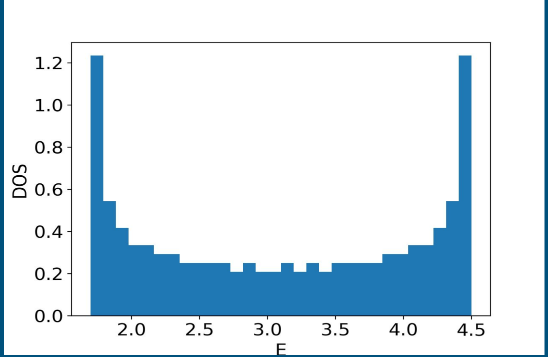
```
4.6619
1.5381
3.8052
1.5407
4.2775
2.3948
4.2367
1.9024
4.6284
1.9841
3.8002
1.6519
4.6593
2.3155
4.5481
1.8632
3.8101
2.0681
4.3173
1.5793
3.8245
2.1717
4.5073
1.7550
4.1530
```



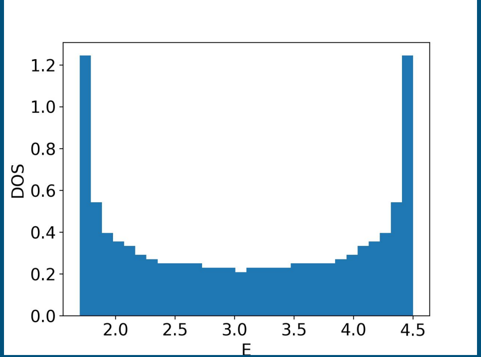
N=128, t=0.7,
E=3.1, 30 bins,
Normalized



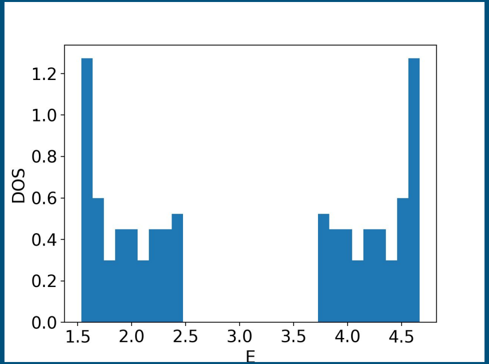
N=256, t=0.7,
E=3.1, 30 bins,
Normalized



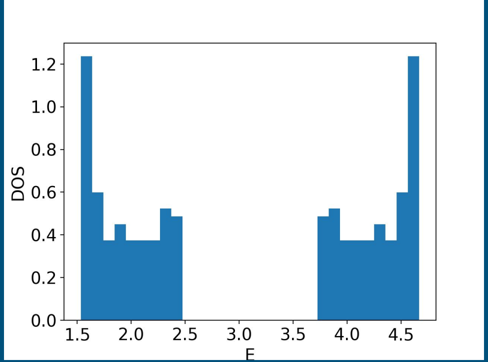
N=512, t=0.7,
E=3.1, 30 bins,
Normalized



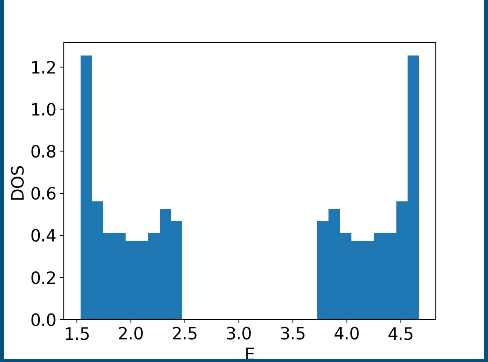
N=1024, t=0.7,
E=3.1, 30 bins,
Normalized



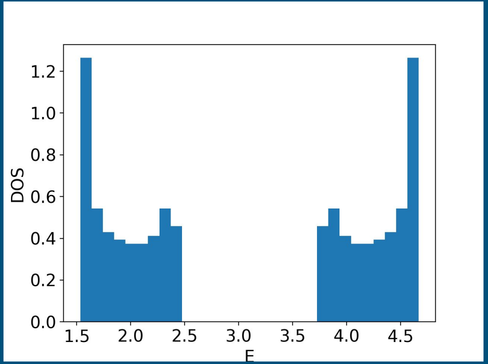
N=128, t=0.7,
E1=3.8, E2=2.4, 30
bins, Normalized



N=256, t=0.7,
E1=3.8, E2=2.4, 30
bins, Normalized



N=512, t=0.7,
E1=3.8, E2=2.4, 30
bins, Normalized

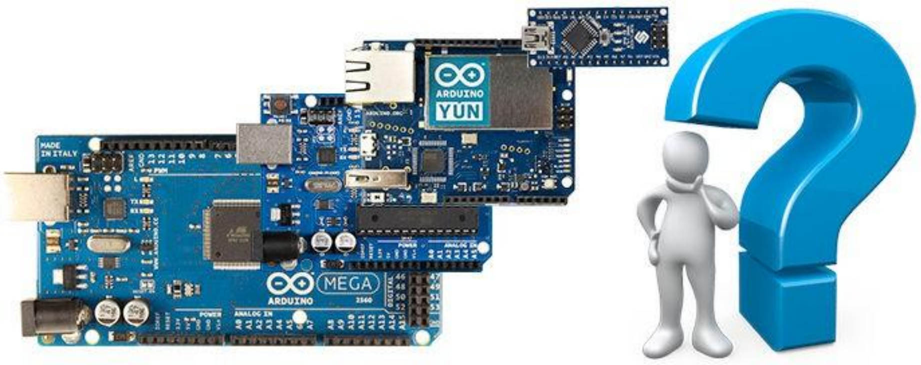


N=1024, t=0.7,
E1=3.8, E2=2.4, 30
bins, Normalized

Temperature and Humidity Sensor

Background Info

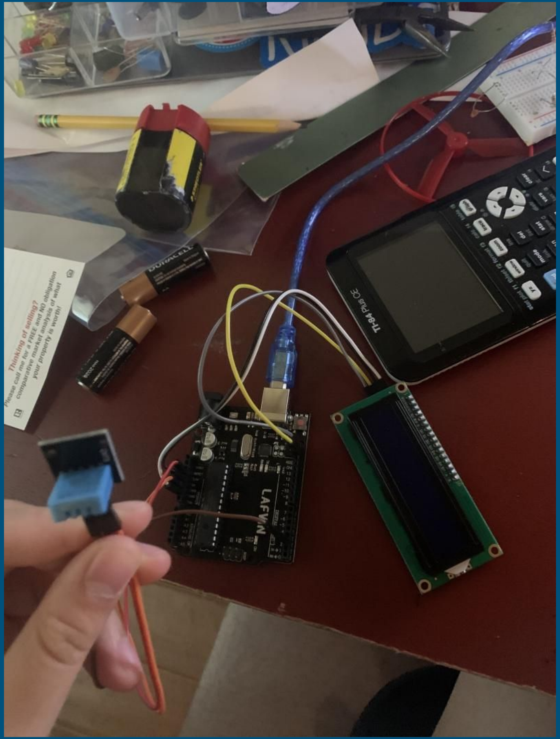
- An arduino is a digital board that executes commands: we can play music or change light colors after we connect it to our computer and run code



Components Used

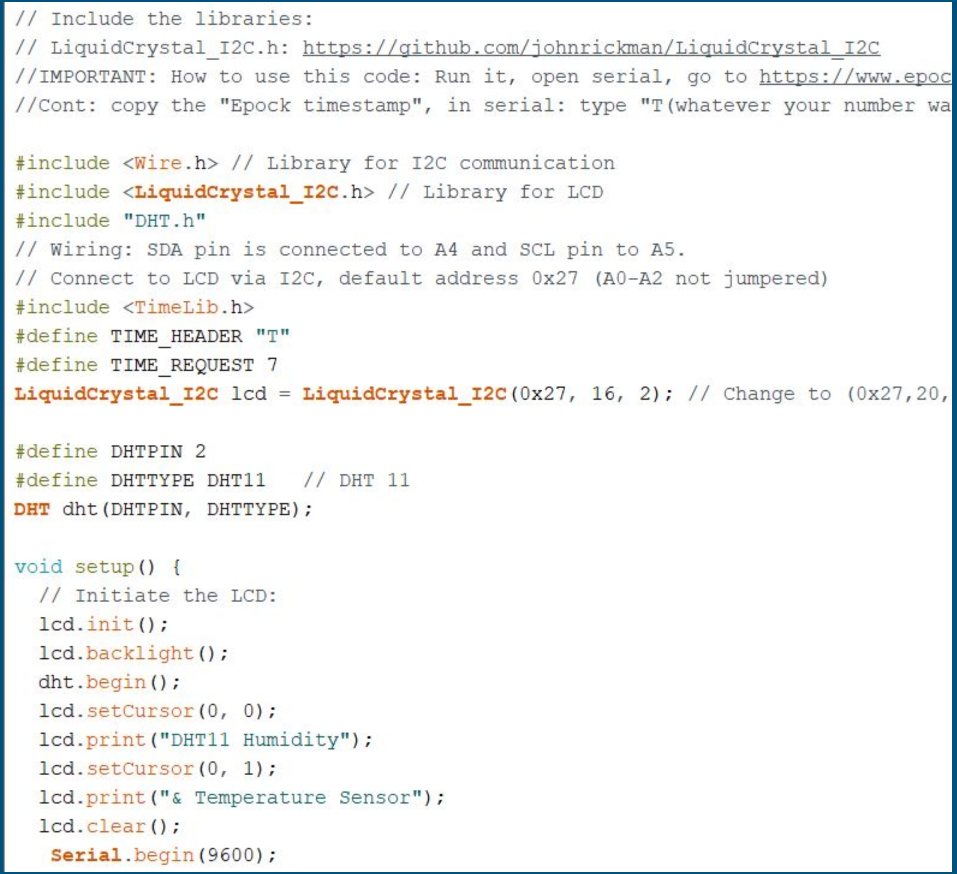
- In our arduino kit, we used the following materials:

01. Wires
02. DHT11 Temperature & humidity sensor
03. Stepper motor
04. LCD display



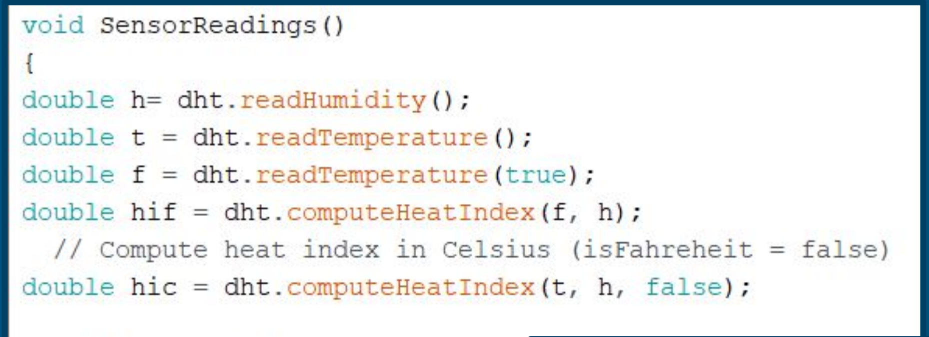
Coding

- Our coding was separated into two major parts: first we coded how to check the temperature and the humidity, then added the time



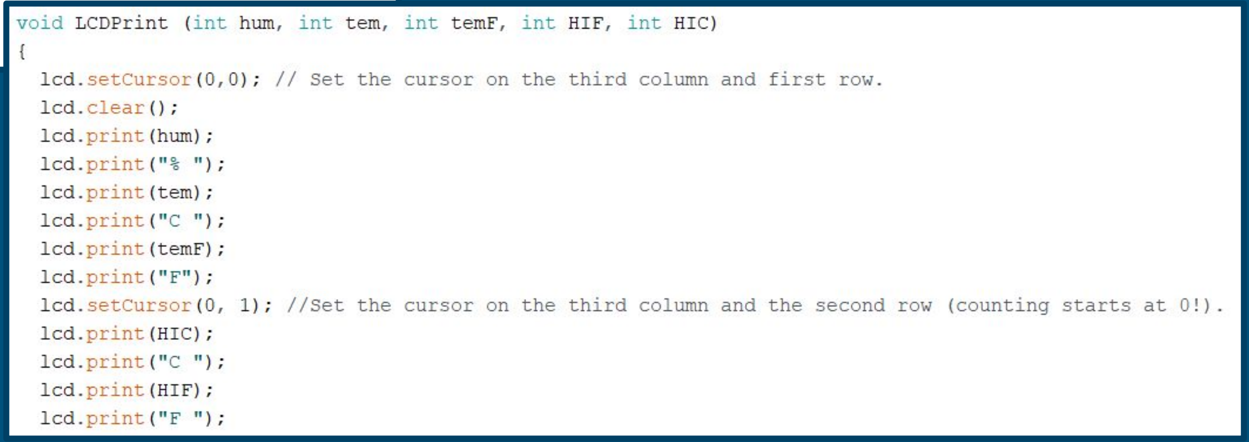
```
// Include the libraries:
// LiquidCrystal_I2C.h: https://github.com/johnrickman/LiquidCrystal_I2C
//IMPORTANT: How to use this code: Run it, open serial, go to https://www.epoc
//Cont: copy the "Epock timestamp", in serial: type "T(whatever your number wa

#include <Wire.h> // Library for I2C communication
#include <LiquidCrystal_I2C.h> // Library for LCD
#include "DHT.h"
// Wiring: SDA pin is connected to A4 and SCL pin to A5.
// Connect to LCD via I2C, default address 0x27 (A0-A2 not jumpered)
#include <TimeLib.h>
#define TIME_HEADER "T"
#define TIME_REQUEST 7
LiquidCrystal_I2C lcd = LiquidCrystal_I2C(0x27, 16, 2); // Change to (0x27,20,

#define DHTPIN 2
#define DHTTYPE DHT11   // DHT 11
DHT dht(DHTPIN, DHTTYPE);

void setup() {
  // Initiate the LCD:
  lcd.init();
  lcd.backlight();
  dht.begin();
  lcd.setCursor(0, 0);
  lcd.print("DHT11 Humidity");
  lcd.setCursor(0, 1);
  lcd.print("& Temperature Sensor");
  lcd.clear();
   Serial.begin(9600);
```

Coding - Temperature/Humidity Sensor



```
void SensorReadings()
{
double h= dht.readHumidity();
double t = dht.readTemperature();
double f = dht.readTemperature(true);
double hif = dht.computeHeatIndex(f, h);
  // Compute heat index in Celsius (isFahreheit = false)
double hic = dht.computeHeatIndex(t, h, false);

LCDPrint(h,t,f,hif,hic);
```

Take Readings from the sensor and store them as variables



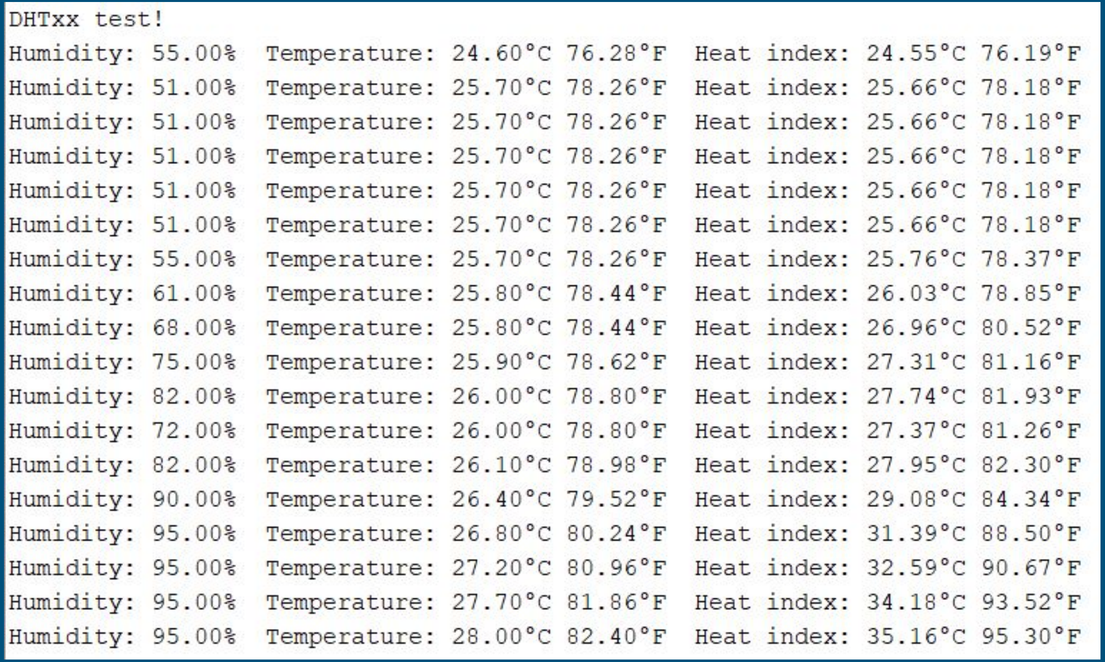
```
void LCDPrint (int hum, int tem, int temF, int HIF, int HIC)
{
  lcd.setCursor(0,0); // Set the cursor on the third column and first row.
  lcd.clear();
  lcd.print(hum);
  lcd.print("% ");
  lcd.print(tem);
  lcd.print("C ");
  lcd.print(temF);
  lcd.print("F");
  lcd.setCursor(0, 1); //Set the cursor on the third column and the second row (counting starts at 0!).
  lcd.print(HIC);
  lcd.print("C ");
  lcd.print(HIF);
  lcd.print("F ");
```

Take the variables and print them on the LCD Screen, then repeat

Testing - Temperature/Humidity Sensor

Started off by just showing it on our computers to test it without the LCD Screen

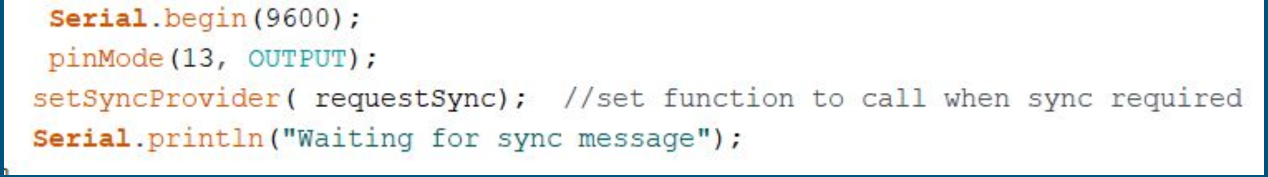
When we blew on the sensor



```
DHTxx test!
Humidity: 55.00%  Temperature: 24.60°C 76.28°F  Heat index: 24.55°C 76.19°F
Humidity: 51.00%  Temperature: 25.70°C 78.26°F  Heat index: 25.66°C 78.18°F
Humidity: 51.00%  Temperature: 25.70°C 78.26°F  Heat index: 25.66°C 78.18°F
Humidity: 51.00%  Temperature: 25.70°C 78.26°F  Heat index: 25.66°C 78.18°F
Humidity: 51.00%  Temperature: 25.70°C 78.26°F  Heat index: 25.66°C 78.18°F
Humidity: 51.00%  Temperature: 25.70°C 78.26°F  Heat index: 25.66°C 78.18°F
Humidity: 55.00%  Temperature: 25.70°C 78.26°F  Heat index: 25.76°C 78.37°F
Humidity: 61.00%  Temperature: 25.80°C 78.44°F  Heat index: 26.03°C 78.85°F
Humidity: 68.00%  Temperature: 25.80°C 78.44°F  Heat index: 26.96°C 80.52°F
Humidity: 75.00%  Temperature: 25.90°C 78.62°F  Heat index: 27.31°C 81.16°F
Humidity: 82.00%  Temperature: 26.00°C 78.80°F  Heat index: 27.74°C 81.93°F
Humidity: 72.00%  Temperature: 26.00°C 78.80°F  Heat index: 27.37°C 81.26°F
Humidity: 82.00%  Temperature: 26.10°C 78.98°F  Heat index: 27.95°C 82.30°F
Humidity: 90.00%  Temperature: 26.40°C 79.52°F  Heat index: 29.08°C 84.34°F
Humidity: 95.00%  Temperature: 26.80°C 80.24°F  Heat index: 31.39°C 88.50°F
Humidity: 95.00%  Temperature: 27.20°C 80.96°F  Heat index: 32.59°C 90.67°F
Humidity: 95.00%  Temperature: 27.70°C 81.86°F  Heat index: 34.18°C 93.52°F
Humidity: 95.00%  Temperature: 28.00°C 82.40°F  Heat index: 35.16°C 95.30°F
```

Coding - Time

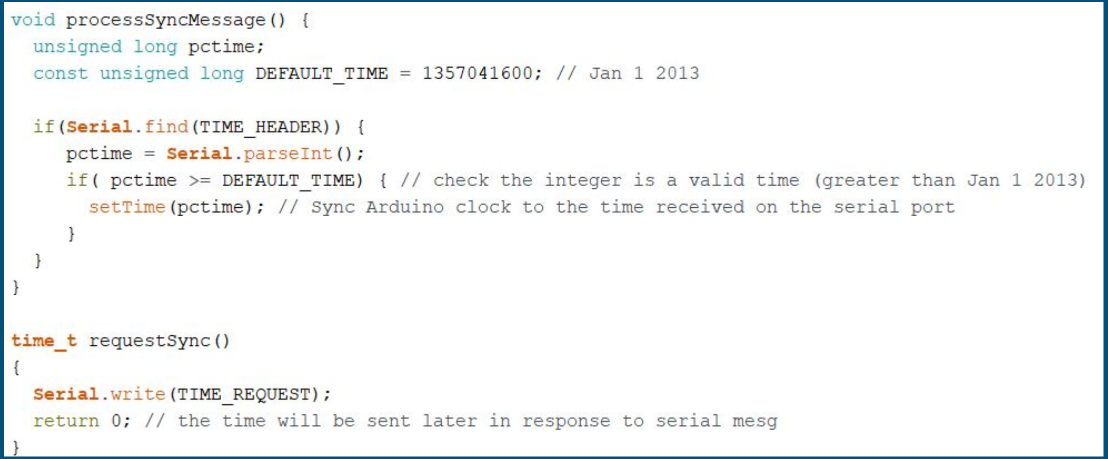
Note: arduino doesn’t have a time connector unless bought seperately



```
 Serial.begin(9600);
 pinMode(13, OUTPUT);
setSyncProvider( requestSync);  //set function to call when sync required
Serial.println("Waiting for sync message");
```

Manually input current time

Counts the time based off the “current time” that we input

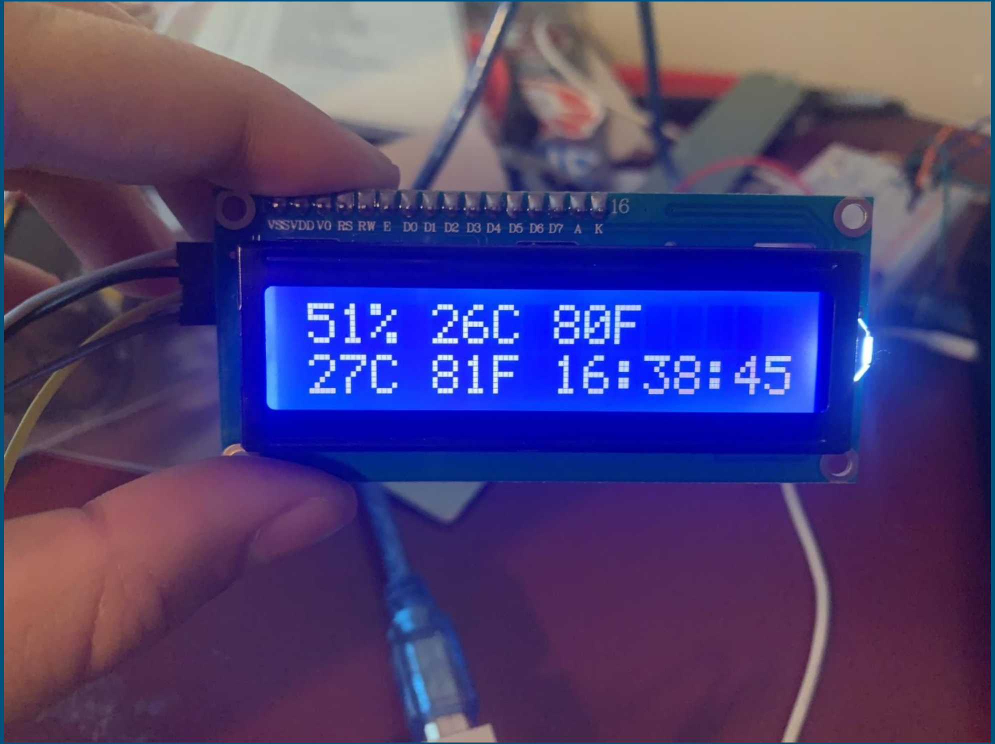


```
void processSyncMessage() {
  unsigned long pctime;
  const unsigned long DEFAULT_TIME = 1357041600; // Jan 1 2013

  if(Serial.find(TIME_HEADER)) {
     pctime = Serial.parseInt();
     if( pctime >= DEFAULT_TIME) { // check the integer is a valid time (greater than Jan 1 2013)
       setTime(pctime); // Sync Arduino clock to the time received on the serial port
     }
  }
}

time_t requestSync()
{
  Serial.write(TIME_REQUEST);
  return 0; // the time will be sent later in response to serial mesg
}
```

Final Product

- After running our code, we were able to display the temperature and humidity of the air around the arduino in addition to the time!
- The time updates every 2 seconds

Future work: every hour passed the time is off by ~2 seconds



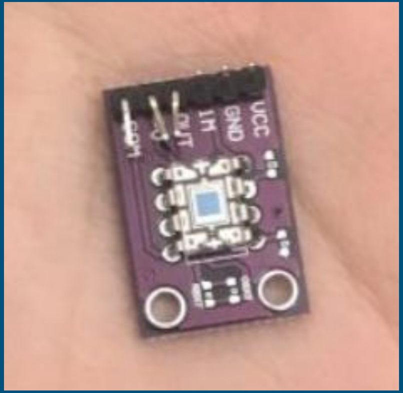
Spectrometer

Extra Project!

- We had spare time, so we decided to make our very own spectrometer!
- What is a spectrometer?
  - It’s an item used in chemistry to measure mixtures and/or solutions through measuring the light absorbance as a function of wavelength -- essentially operates like a prism
  - By measuring the level of absorbance in the liquid, we can use this to identify liquids
  - There are many types of spectrometers!

Light Sensor

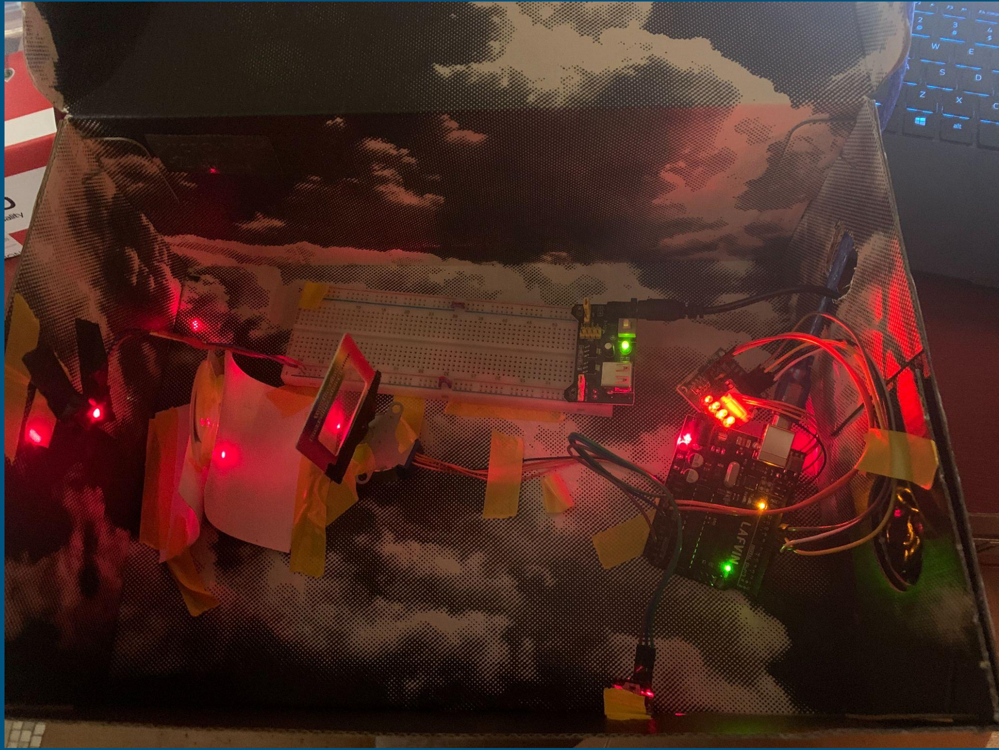
- Our Arduino kit didn't come with a normal light sensor, so we had to solder one on our own
- Soldering: connecting parts of the circuit through melting an alloy in between two metals in order to get good electrical contact



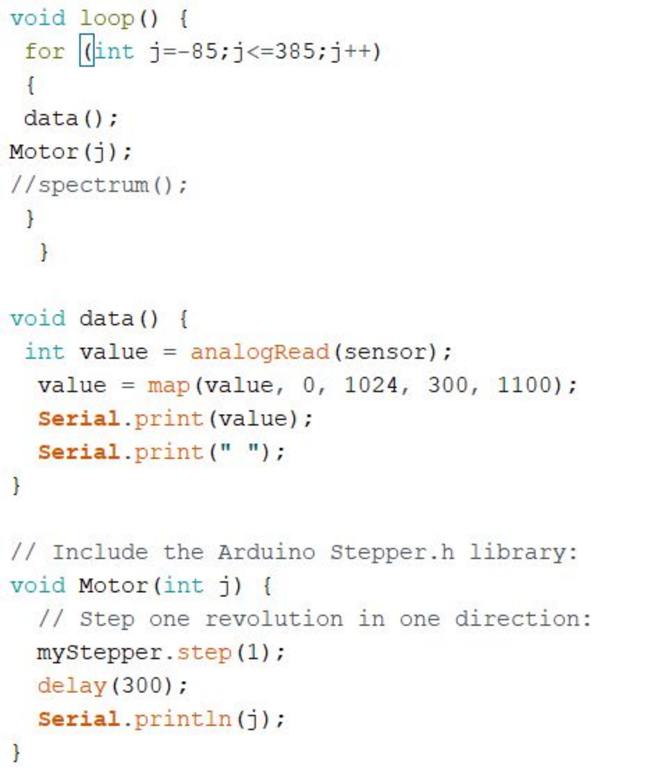
The light sensor we soldered

Components Used

01. Cardboard box
02. Stepper motor + its driver module
03. Light sensor (soldered)
04. Breadboard + 5V power supply
05. Lense
06. Laser
07. Arduino



Code - Receiving Data

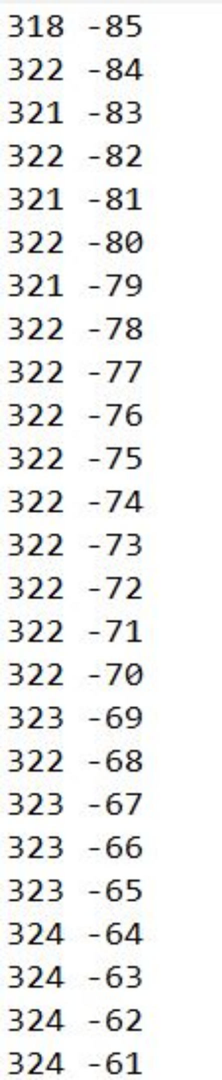


```
void loop() {
 for (int j=-85;j<=385;j++)
 {
 data();
Motor(j);
//spectrum();
 }
  }

void data() {
 int value = analogRead(sensor);
  value = map(value, 0, 1024, 300, 1100);
  Serial.print(value);
  Serial.print(" ");
}

// Include the Arduino Stepper.h library:
void Motor(int j) {
  // Step one revolution in one direction:
  myStepper.step(1);
  delay(300);
  Serial.println(j);
}
```

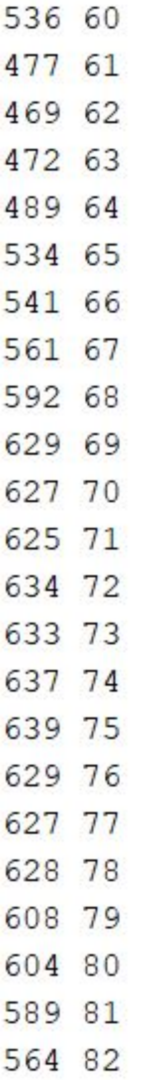
For loop: tells the computer to run the code within a certain amount of steps

Data: receives and prints the intensity of light

Motor: tells the motor to spin a step at a time

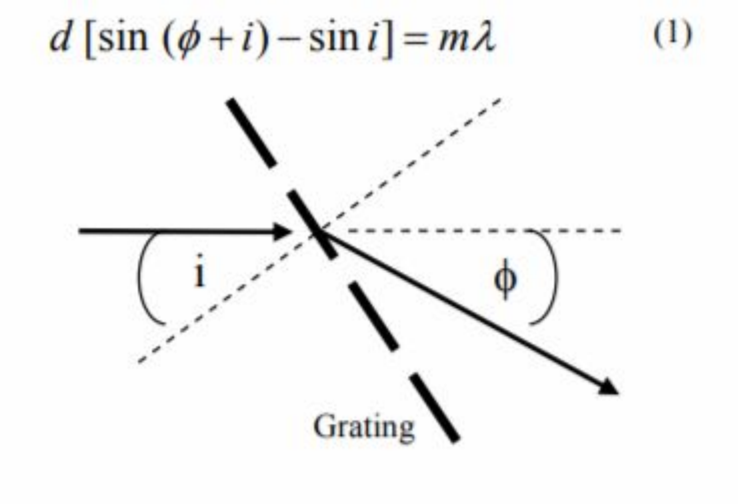


```
318 -85
322 -84
321 -83
322 -82
321 -81
322 -80
321 -79
322 -78
322 -77
322 -76
322 -75
322 -74
322 -73
322 -72
322 -71
322 -70
323 -69
322 -68
323 -67
323 -66
323 -65
324 -64
324 -63
324 -62
324 -61
```



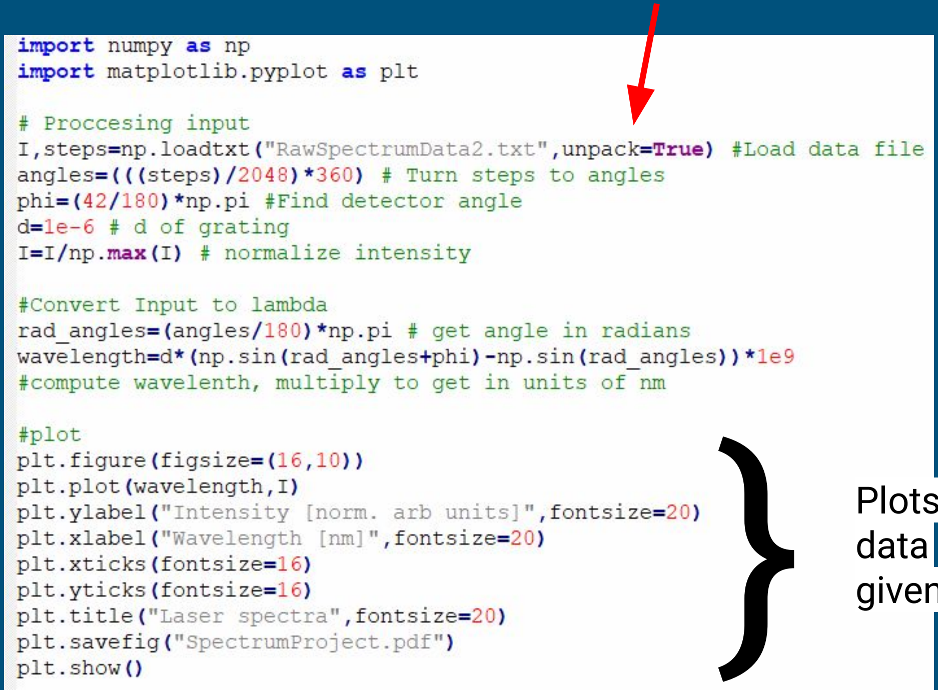
```
536 60
477 61
469 62
472 63
489 64
534 65
541 66
561 67
592 68
629 69
627 70
625 71
634 72
633 73
637 74
639 75
629 76
627 77
628 78
608 79
604 80
589 81
564 82
```

Code - Analyzing Data



$$d\,[\sin(\phi + i) - \sin i] = m\lambda \qquad (1)$$

Diagram of Spectrometer



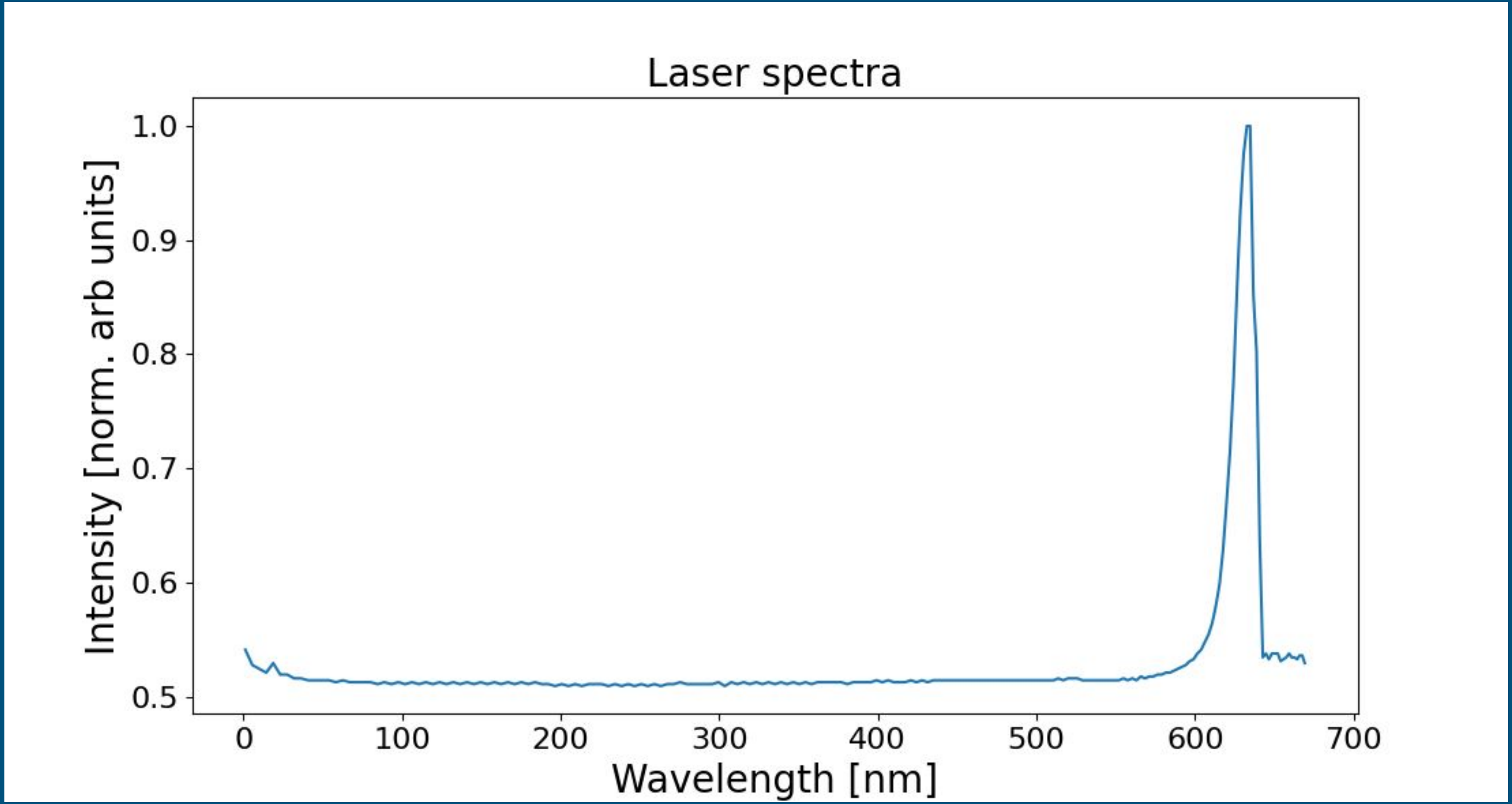
```
import numpy as np
import matplotlib.pyplot as plt

# Proccesing input
I,steps=np.loadtxt("RawSpectrumData2.txt",unpack=True) #Load data file
angles=(((steps)/2048)*360) # Turn steps to angles
phi=(42/180)*np.pi #Find detector angle
d=1e-6 # d of grating
I=I/np.max(I) # normalize intensity

#Convert Input to lambda
rad_angles=(angles/180)*np.pi # get angle in radians
wavelength=d*(np.sin(rad_angles+phi)-np.sin(rad_angles))*1e9
#compute wavelenth, multiply to get in units of nm

#plot
plt.figure(figsize=(16,10))
plt.plot(wavelength,I)
plt.ylabel("Intensity [norm. arb units]",fontsize=20)
plt.xlabel("Wavelength [nm]",fontsize=20)
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
plt.title("Laser spectra",fontsize=20)
plt.savefig("SpectrumProject.pdf")
plt.show()
```

Plots data given

Results

Results

Laser Data: peak at 670 nm



Sources

01. https://github.com/johnrickman/LiquidCrystal_I2C

02. https://stackoverflow.com/questions/59815331/how-to-use-time-library-in-arduino

03. https://www.instructables.com/Spectrometer-Using-Arduino/

04. https://www.youtube.com/watch?v=Wfp58H6U60g

Thank you!